

What Is Claimed Is:

1 1. A method for generating code to perform anticipatory prefetching
2 for data references, comprising:
3 receiving code to be executed on a computer system;
4 analyzing the code to identify data references to be prefetched, wherein
5 analyzing the code involves,
6 performing a first marking phase in which only data
7 references located in blocks that are certain to execute are
8 considered in determining which data references are covered by
9 preceding data references, and
10 performing a second marking phase in which data
11 references that are located in blocks that are not certain to execute
12 are considered; and
13 inserting prefetch instructions into the code in advance of the identified
14 data references.

1 2. The method of claim 1, further comprising:
2 profiling execution of the code to produce profiling results; and
3 using the profiling results to determine whether a given block of
4 instructions is executed frequently enough to perform the second marking phase
5 on the given block of instructions.

1 3. The method of claim 2, wherein determining whether the given
2 block of instructions is executed frequently enough to perform the second
3 marking phase involves comparing a frequency of execution for the given block
4 from the profiling results with a threshold value indicating a minimum frequency
5 of execution to be considered in the second marking phase.

1 4. The method of claim 1, wherein analyzing the code involves:
2 identifying loop bodies within the code; and
3 identifying data references to be prefetched from within the loop bodies.

1 5. The method of claim 4, wherein if there exists a nested loop within
2 the code, analyzing the code involves:
3 examining an innermost loop in the nested loop; and
4 examining a loop outside the innermost loop if the innermost loop is
5 smaller than a minimum size or is executed fewer than a minimum number of
6 iterations.

1 6. The method of claim 4, wherein analyzing the code to identify data
2 references to be prefetched involves examining a pattern of data references over
3 multiple loop iterations.

1 7. The method of claim 1, wherein analyzing the code involves
2 analyzing the code within a compiler.

1 8. A computer-readable storage medium storing instructions that
2 when executed by a computer cause the computer to perform a method for
3 generating code to perform anticipatory prefetching for data references, the
4 method comprising:

5 receiving code to be executed on a computer system;
6 analyzing the code to identify data references to be prefetched, wherein
7 analyzing the code involves,
8 performing a first marking phase in which only data
9 references located in blocks that are certain to execute are

10 considered in determining which data references are covered by
11 preceding data references, and
12 performing a second marking phase in which data
13 references that are located in blocks that are not certain to execute
14 are considered; and
15 inserting prefetch instructions into the code in advance of the identified
16 data references.

1 9. The computer-readable storage medium of claim 8, wherein the
2 method further comprises:

3 profiling execution of the code to produce profiling results; and
4 using the profiling results to determine whether a given block of
5 instructions is executed frequently enough to perform the second marking phase
6 on the given block of instructions.

1 10. The computer-readable storage medium of claim 9, wherein
2 determining whether the given block of instructions is executed frequently enough
3 to perform the second marking phase involves comparing a frequency of
4 execution for the given block from the profiling results with a threshold value
5 indicating a minimum frequency of execution to be considered in the second
6 marking phase.

7
1 11. The computer-readable storage medium of claim 8, wherein
2 analyzing the code involves:
3 identifying loop bodies within the code; and
4 identifying data references to be prefetched from within the loop bodies.

1 12. The computer-readable storage medium of claim 11, wherein if
2 there exists a nested loop within the code, analyzing the code involves:
3 examining an innermost loop in the nested loop; and
4 examining a loop outside the innermost loop if the innermost loop is
5 smaller than a minimum size or is executed fewer than a minimum number of
6 iterations.

1 13. The computer-readable storage medium of claim 11, wherein
2 analyzing the code to identify data references to be prefetched involves examining
3 a pattern of data references over multiple loop iterations.

1 14. The computer-readable storage medium of claim 11, wherein
2 analyzing the code involves analyzing the code within a compiler.

1 15. An apparatus that generates code to perform anticipatory
2 prefetching for data references, comprising:
3 a receiving mechanism that is configured to receive code to be executed
4 on a computer system;
5 an analysis mechanism that is configured to analyze the code to identify
6 data references to be prefetched, wherein the analysis mechanism is configured to,
7 perform a first marking phase in which only data references
8 located in blocks that are certain to execute are considered in
9 determining which data references are covered by preceding data
10 references, and to
11 perform a second marking phase in which data references
12 that are located in blocks that are not certain to execute are
13 considered; and

14 an insertion mechanism that is configured to insert prefetch instructions
15 into the code in advance of the identified data references.

1 16. The apparatus of claim 15, further comprising a profiling
2 mechanism that is configured to profile execution of the code to produce profiling
3 results;

4 wherein the analysis mechanism is configured to use the profiling results
5 to determine whether a given block of instructions is executed frequently enough
6 to perform the second marking phase on the given block of instructions.

1 17. The apparatus of claim 16, wherein the analysis mechanism is
2 configured to compare a frequency of execution for the given block from the
3 profiling results with a threshold value indicating a minimum frequency of
4 execution to be considered in the second marking phase.

1 18. The apparatus of claim 15, wherein the analysis mechanism is
2 configured to:

3 identify loop bodies within the code; and to
4 identify data references to be prefetched from within the loop bodies.

1 19. The apparatus of claim 18, wherein if there exists a nested loop
2 within the code, the analysis mechanism is configured to:
3 examine an innermost loop in the nested loop; and to
4 examine a loop outside the innermost loop if the innermost loop is smaller
5 than a minimum size or is executed fewer than a minimum number of iterations.

1 20. The apparatus of claim 18, wherein the analysis mechanism is
2 configured to examine a pattern of data references over multiple loop iterations.

1 21. The apparatus of claim 15, wherein the apparatus resides within a
2 compiler.

1 22. A method for generating code to perform anticipatory prefetching
2 for data references, comprising:

3 receiving code to be executed on a computer system;

4 analyzing the code to identify data references to be prefetched, wherein
5 analyzing the code involves,

6 examining an array reference made through an array
7 subscript,

8 determining a function for the array subscript in terms of a
9 loop index,

10 using the function to calculate a difference between array
11 indexes for consecutive loop iterations, and

12 considering the array reference as a candidate for
13 prefetching if the difference between array indexes for consecutive
14 loop iterations is a constant value; and

15 inserting prefetch instructions into the code in advance of the identified
16 data references.

1 23. The method of claim 22, wherein determining the function for the
2 array subscript in terms of a loop index involves chasing down data dependencies
3 associated with the array subscript if such data dependencies exist.

1 24. The method of claim 22, wherein the array reference is considered
2 a candidate for prefetching if the difference between array indexes is a constant
3 value for some but not all consecutive loop iterations.

1 25. The method of claim 24, wherein the array reference is considered
2 a candidate for prefetching if the difference between array indexes depends on a
3 modulo operator that causes the difference between array indexes to occasionally
4 vary from the constant value.

1 26. The method of claim 22, wherein analyzing the code involves:
2 identifying loop bodies within the code; and
3 identifying data references to be prefetched from within the loop bodies.

1 27. The method of claim 26, wherein if there exists a nested loop
2 within the code, analyzing the code involves:
3 examining an innermost loop in the nested loop; and
4 examining a loop outside the innermost loop if the innermost loop is
5 smaller than a minimum size or is executed fewer than a minimum number of
6 iterations.
7

1 28. The method of claim 26, wherein analyzing the code involves
2 examining a pattern of data references over multiple loop iterations.

1 29. The method of claim 22, wherein analyzing the code involves
2 analyzing the code within a compiler.

1 30. A computer-readable storage medium storing instructions that
2 when executed by a computer cause the computer to perform a method for
3 generating code to perform anticipatory prefetching for data references, the
4 method comprising:
5 receiving code to be executed on a computer system;

1 analyzing the code to identify data references to be prefetched, wherein
2 analyzing the code involves,
3 examining an array reference made through an array
4 subscript,
5 determining a function for the array subscript in terms of a
6 loop index,
7 using the function to calculate a difference between array
8 indexes for consecutive loop iterations, and
9 considering the array reference as a candidate for
10 prefetching if the difference between array indexes for consecutive
11 loop iterations is a constant value; and
12 inserting prefetch instructions into the code in advance of the identified
13 data references.

1 31. The computer-readable storage medium of claim 30, wherein
2 determining the function for the array subscript in terms of a loop index involves
3 chasing down data dependencies associated with the array subscript if such data
4 dependencies exist.

1 32. The computer-readable storage medium of claim 30, wherein the
2 array reference is considered a candidate for prefetching if the difference between
3 array indexes is a constant value for some but not all consecutive loop iterations.

1 33. The computer-readable storage medium of claim 32, wherein the
2 array reference is considered a candidate for prefetching if the difference between
3 array indexes depends on a modulo operator that causes the difference between
4 array indexes to occasionally vary from the constant value.

1 34. The computer-readable storage medium of claim 30, wherein
2 analyzing the code involves:
3 identifying loop bodies within the code; and
4 identifying data references to be prefetched from within the loop bodies.

1 35. The computer-readable storage medium of claim 34, wherein if
2 there exists a nested loop within the code, analyzing the code involves:
3 examining an innermost loop in the nested loop; and
4 examining a loop outside the innermost loop if the innermost loop is
5 smaller than a minimum size or is executed fewer than a minimum number of
6 iterations.
7

1 36. The computer-readable storage medium of claim 34, wherein
2 analyzing the code involves examining a pattern of data references over multiple
3 loop iterations.

1 37. The computer-readable storage medium of claim 30, wherein
2 analyzing the code involves analyzing the code within a compiler.

1 38. An apparatus that generates code to perform anticipatory
2 prefetching for data references, comprising:
3 a receiving mechanism that is configured to receive code to be executed
4 on a computer system;
5 an analysis mechanism that is configured to analyze the code to identify
6 data references to be prefetched, wherein the analysis mechanism is configured to,
7 examine an array reference made through an array
8 subscript,
9 determine a function for the array subscript in terms of a

1 loop index,
2 use the function to calculate a difference between array
3 indexes for consecutive loop iterations, and to
4 consider the array reference as a candidate for prefetching
5 if the difference between array indexes for consecutive loop
6 iterations is a constant value; and
7 an insertion mechanism that is configured to insert prefetch instructions
8 into the code in advance of the identified data references.

1 39. The apparatus of claim 38, wherein while determining the function
2 for the array subscript in terms of a loop index, the analysis mechanism is
3 configured to chase down data dependencies associated with the array subscript if
4 such data dependencies exist.

1 40. The apparatus of claim 38, wherein the analysis mechanism is
2 configured to consider the array reference as a candidate for prefetching if the
3 difference between array indexes is a constant value for some but not all
4 consecutive loop iterations.

1 41. The apparatus of claim 40, wherein the analysis mechanism is
2 configured to consider the array reference as a candidate for prefetching if the
3 difference between array indexes depends on a modulo operator that causes the
4 difference between array indexes to occasionally vary from the constant value.

1 42. The apparatus of claim 38, wherein the analysis mechanism is
2 configured to:
3 identify loop bodies within the code; and to
4 identify data references to be prefetched from within the loop bodies.

1 43. The apparatus of claim 42, wherein if there exists a nested loop
2 within the code, the analysis mechanism is configured to:
3 examine an innermost loop in the nested loop; and to
4 examine a loop outside the innermost loop if the innermost loop is smaller
5 than a minimum size or is executed fewer than a minimum number of iterations.

1 44. The apparatus of claim 42, wherein the analysis mechanism is
2 configured to analyze a pattern of data references over multiple loop iterations.

1 45. The apparatus of claim 38, wherein the apparatus resides within a
2 compiler.